```
=> d his


        (FILE 'HOME' ENTERED AT 14:17:51 ON 20 NOV 2001)

        FILE 'USPATFULL' ENTERED AT 14:18:11 ON 20 NOV 2001
L1          302 S (REGISTERING OR REGISTRY)(P)(NOTIFICATION#)
                SET HIGH OFF
L2         6393 S OBJECT ORIENT?
                SET HIGH ON
L3           67 S L1 AND L2
L4            1 S SPRINGMEYER, STEVE?/IN OR HASHA, RICHARD?/IN
```

TI   Platform independent distributed management system for manipulating
        managed objects in a network
PI   US 6282568   B1   20010828
DETD   In addition, the Platform class 500 has five specific methods to
        register for various events that emanate from the MIS. The MIS uses a
        registry scheme in which users that are interested in an event
        register with the MIS. When the event occurs, the MIS sends a
        notification to the registered users. Three of these methods are
        most commonly used, namely an attribute value change method for changes
        in attributes of a managed object, an object creation method and an
        object deletion method . The three methods shown below add a
        registration for these events to the MIS event **registry** and
        allow the user to be notified when such an event occurs. The user is
        expected to provide a "listener" for these events which implements an
        interface defined by the IEventReportListener class. This interface
        specifies a handler method which a listener object must implement. More
        details are provided below in the description of the interface
        IEventReportListener.
DETD   Similar to methods for **registering** of events in the case of a
        MOHandle object being included or excluded from the collection, the
        following methods allow user to de-register for such event
        notifications.
CLM   What is claimed is:
        17. A method according to claim 14 wherein step (b) comprises the steps
        of: (b2) instantiating an event dispatcher object for receiving
        notifications from the management information server; and (b3)
        registering each handle object with the event dispatcher object
        with an ID to receive the **notifications**.

TI   Using query language for provider and subscriber registrations
PI   (US 6275957)   B1   20010814
AB   Systems and methods for reporting the occurrence of events in a
computer
        system to event subscriber software. A computer system includes a
        central repository wherein event subscribers register the types or
        classes of events for which they require **notification** and
        event providers register the types or classes of events they are
capable
        of detecting and for which they will provide **notifications**.
        The registrations, both by providers and subscribers, are made
according
        to a standardized hierarchical classification of event classes and are
        preferably expressed in the form of queries. The computer system also
        includes an event-filtering component that receives **notification**
        of the occurrence of events, filters the events, and reports selected
        events to the subscriber software. The event-filtering component can
        expose standardized interfaces to the event providers that report
events
        and to the subscriber software to which events are reported. Filtering
        can be facilitated by event-filtering definitions written in a query
        language and associated with the subscriber software. The definitions
        are processed in the context of an object-oriented, hierarchical
        classification of event classes that comprehend any possible events
that
        can be reported by the event providers. When reported events satisfy
one
        or more query-based filtering definitions, the events are passed to the
        appropriate subscriber software. Otherwise, the events are discarded.
        Events can be filtered and grouped according to the time of their
        occurrence. Filtering can be further simplified by **registering**
        event-reporting definitions defining the scope of events to be reported
        by particular event providers.
CLM   What is claimed is:
        1. In a computer system having one or more event providers capable of
        detecting the occurrence of certain conditions or events in said
        computer system or the environment of said computer system and having
        one or more event subscribers requiring **notification** of
        certain conditions or events in said computer system or the environment
        of said computer system, a method for providing an interface between
        said event providers and said event subscribers comprising: the step of
        defining a hierarchical classification of events that comprehend a set
        of possible conditions and events; the step of providing to the
computer
        system an event filtering and reporting component; the step of
        registering with said event filtering and reporting component
        (a) an event provider definition associated with each event provider,
        wherein the event provider definition is expressed in terms of said
        hierarchical classification of events and specifies the conditions or
        events of which such event provider will provide **notification**
        to the event filtering and reporting component and (b) an event
        subscriber definition associated with each said event subscriber,
        wherein the event subscriber definition is expressed in terms of said
        hierarchical classification of events and specifies the conditions or
        events for which such event subscriber requests **notification**
        from the event filtering and reporting component; the step of
filtering,
        by the event filtering and reporting component, conditions or events
        reported by the event providers to the event filtering and reporting

component; and the step of notifying each said event subscriber of only those conditions or events that satisfy the event subscriber definition associated with that event subscriber.

3. The method of claim 2 further comprising, after the **registering** step: the step of detecting, by the event providers registered with the event filtering and reporting component, the occurrence of a condition or event in the computer or the environment of

the computer that satisfies the event provider definition for that event

provider; and the step of sending **notification** of the occurrence of said condition or event from the event provider to the event filtering and reporting component.

5. In a computer system having one or more event providers capable of detecting the occurrence of certain conditions or events in said computer system or the environment of said computer system and having one or more event subscribers requiring **notification** of certain conditions or events in said computer system or the environment of said computer system, a system for providing an interface between said event providers and said event subscribers comprising: means for defining a hierarchical classification of events that comprehend a set of possible conditions and events; means, logically connected to said event providers and said event subscribers, for filtering conditions or events detected by said event providers and for reporting certain conditions or events to said event subscribers; and means for **registering** with said event filtering and reporting means (a) an event provider definition associated with each event provider, wherein the event provider definition is expressed in terms of said hierarchical

classification of events and specifies the conditions or events of which

such event provider will provide **notification** to the event filtering and reporting means and (b) an event subscriber definition associated with each said event subscriber, wherein the event subscriber

definition is expressed in terms of said hierarchical classification of events and specifies the conditions or events for which such event subscriber requests **notification** from the event filtering and reporting means, and wherein said event filtering and reporting means filters the conditions or events reported by the event providers and notifies each said event subscriber of only those conditions or events that satisfy the event subscriber definition associated with that event subscriber.

9. An article of manufacture for use in a computer system having a CPU, o

TI   Automatic updating of diverse software products on multiple client
      computer systems by downloading scanning application to client computer
      and generating software list on client computer
PI   US 6151643   20001121
DETD Activity types not represented in the example above include Undo of
      Updates by the recovery module 908, **registering** for service,
      and **registering** for **notification** for updates to
      specific products.

TI   Integrated three-tier application framework with automated class and table generation

PI   US 6085198   20000704

DETD   Server-side update management component 304B acts in association with change management component 302B to ensure that notices regarding changes to data are transmitted to all interested elements of the system. Under the active **notification** scheme using interest objects, update management component 304B maintains a **registry** of clients and other servers that wish to be notified when a designated data object is changed. When a change is made, interested elements within application server 307 and interested clients receive **notification** of the change. **Notification** within each client is typically resolved by the respective client-side update management component 304A.

DETD   As one of the steps in the change transaction, update management component 304B is informed of the change, as indicated by arrow 807. Interested clients are identified by update management component 304B, and a notice of the change is transmitted to the update management component 304A of the interested clients, as indicated by arrow 812. The client update management component 304A then determines, based on an interest **registry**, which elements on the client should receive **notification** of the change. The data objects in object cache 303A are typically updated at the time the client is notified of the change.

TI   Method of transmitting a notification to a receiver from plural
       notification services in a distributed application network, and a
       network for implementing the method
PI   (US 6073184   20000606
SUMM   firstly enrolling the second **notification** service as a
       receiver of **notifications** from said first **notification**
       service and/or **registering** the first **notification**
       service as an emitter of **notifications** to the second
    notification service; and

SUMM   the step of enrolling the second **notification** service as a
       receiver of **notifications** from said first **notification**
       service and/or of **registering** the first **notification**
       service as an emitter of **notifications** to the second
    notification service is performed on a **notification**
       service administrator making a request to the first **notification**
       service;

SUMM   the step of **registering** the first **notification**
       service as an emitter of **notifications** to the second
    notification service includes the following successive steps:

SUMM   the step of enrolling the second **notification** service as a
       receiver of **notifications** from said first **notification**
       service and/or of **registering** the first **notification**
       service as a transmitter of **notifications** to the second
    notification service is performed by a **notification**
       service administrator making a request to the second
    notification service;

SUMM   the step of **registering** the first **notification**
       service as an emitter of **notifications** to the second
    notification service includes the following successive steps:

SUMM   each **notification** service comprises a set of software
       components each including at least one **notification** channel
       specific to a determined category of **notifications**, and the
       step of **registering** the first **notification** service
       as an emitter of **notifications** to the second
    notification service is performed, while picking up said
    notification, by the **notification** channel of the
       second **notification** service corresponding to the determined
       category of **notifications** implementing the identifier of the
    notification channel associated with the first
    notification service;

SUMM   The invention also provides a distributed-application information
       processing network comprising firstly a first **notification**
       zone having a first **notification** service associated therewith
       and having an emitter linked thereto, and secondly a second
    notification zone having a second **notification** service
       associated therewith and having a receiver linked thereto, each
    notification service including means for notifying
    notifications of a determined category to receivers enrolled
       with the **notification** service and/or for picking up
    notifications of a determined category from emitters registered
       with the **notification** service, the network being characterized
       in that it includes means for enrolling the second **notification**
       service as a receiver of **notifications** from said first

notification service and/or for **registering** the first
notification service as an emitter of **notification** to
the second **notification** service, and means for transmitting
said **notification** from said emitter to said receiver via the
first **notification** service and the second **notification**
service in succession.

DETD   The **notification** server administrator 30 includes an IDL
interface for **registering** emitter objects using the
administrator of the **notification** server. This has the
reference 30A. This interface is adapted to enable emitter objects
desiring to emit **notifications** via the **notification**
service 16 to be registered. In particular, the registration interface
30A addresses to the emitter object seeking registration the reference
of the object forming the **notification** channel with which the
emitter object desires to be registered in order to be able to
broadcast
notifications thereby.

DETD   In addition, when the **notification** service administrator is
adapted to address federation requests to the second
notification service 18, the step of **registering** the
first **notification** service as an emitter of
notifications to the second **notification** service
includes successive steps consisting firstly in the second
notification service 18 addressing an enrollment message to the
first **notification** service 16, and then the first
notification service 16 returning an identifier of the first
notification service to the second **notification**
s

TI   Object-oriented tool for **registering** objects for observation
      and causing **notifications** to be made in the event changes are
      made to an object which is being observed
PI   US 5991536   19991123
TI   Object-oriented tool for **registering** objects for observation
      and causing **notifications** to be made in the event changes are
      made to an object which is being observed
DETD FIG. 4 is a block diagram illustrating the class definitions used to
      implement the present invention. The observed objects 112 in the object
      hierarchy 114 are instances of a BaseNotifier class 400, which inherits
      from a IStandardNotifier class 402. The IStandardNotifier class 402 is
      defined in IBM's ICLUI class library. The BaseNotifier class 400
      provides the functions necessary for notifying the **notification**
      manager 110 when a change is made to an observed object 112 and for
      identifying the observer objects 116 that are registered with the
      observed object 112. The BaseNotifier class 400 supplements these
      functions with functions for adding (**registering**) and removing
      (de-**registering**) observer objects 116 with observed objects
      112.
CLM   What is claimed is:
      1. A method of **notification** in an object-oriented system,
      comprising the steps of: (a) storing an object hierarchy in a computer,
      wherein the object hierarchy includes one or more observed objects; (b)
      registering an observer object with a **notification**
      manager so that the observer object is notified when changes are made
to
      the observed objects; (c) determining when a change is made to one of
      the observed objects in the object hierarchy stored in the computer;
(d)
      informing the **notification** manager of the change made to the
      observed object; (e) notifying the observer object via the
      notification manager of the change made to one of the observed
      objects in the object hierarchy stored in the computer, wherein the
      observed object has no knowledge of the observer object.

      13. The apparatus of claim 8, wherein the **notification** manager
      means further comprises means for **registering** the observer
      objects with the observed objects, so that the observer objects are
      notified when changes are made to the observed objects.

      16. An article of manufacture comprising a program storage medium
      readable by a computer having a memory, the medium tangibly embodying
      one or more programs of instructions executable by the computer to
      perform method steps for **notification** in an object-oriented
      system, the method comprising the steps of: (a) storing an object
      hierarchy in the computer, wherein the object hierarchy includes one or
      more observed objects; (b) **registering** an observer object with
      a **notification** manager so that the observer object is notified
      when changes are made to the observed objects; (c) determining when a
      change is made to one of the observed objects in the object hierarchy
      stored in the computer; (d) informing the **notification** manager
      of the change made to the observed object; (e) notifying an observer
      object of the change made to one of the observed objects in the object
      hierarchy stored in the computer, wherein the observed object has no
      k

TI   Client-server animation system for managing interactive user interface
        characters
PI   US 5983190   19991109
DETD   Once the server is started, the client continues with the process of
        attaching by **registering** a **notification** interface
        with the server. The **notification** interface is used by the
        server whenever it needs to communicate either events or state changes
        with its connected clients. **Notifications** from the server to
        connected clients usually occur on a separate thread of execution in
the
        server. This is necessary in order to prevent any single client from
        blocking the server while it is processing a **notification**.
CLM   What is claimed is:
        6. The method of claim 1 wherein the step of creating an instance of
the
        character comprises: starting execution of the server in response to a
        request from the client; in the server, **registering** a
     notification interface for the client in response to a request
        from the client; and in the server, receiving from the client a request
        telling the server which character to create.

L3   ANSWER 34 OF 67   USPATFULL
TI   Event notification in a computer system
PI   US 5925108   19990720
AB   A system and method separate the order in which event handlers register
      from the order in which the event handlers are notified of events. This
      allows any convenient registration order to be used together with a
      notification order that corresponds to a network architecture, a
      memory hierarchy, or another familiar scale. The **notification**
      order is determined by the event producers, and therefore may be
      reversed without re-**registering** the event handlers. Events may
      be broadcast, may carry data between event handlers, and may be
consumed
      to prevent further **notifications**.
SUMM Distribution according to the order of destination registration may
      result in a First Registered First Notified order or a Last Registered
      First Notified order. In either case, the order in which different
      destinations receive a given message depends on the order in which the
      destinations registered their interest in such messages with some
      central **registry**. In situations where the desired order of
      message receipt does not match the most convenient or established order
      of registration, tying the event **notification** order so closely
      to the registration order is not advantageous.

SUMM Each registration of a destination for event **notification**
      includes an indication of the desired **notification** order of
      the event handler (also termed herein the event "consumer", although
      consumption of an event to prevent further **notifications** is
      optional). This **notification** order does not change after the
      registration. This distinguishes the invention from executive task
      handlers which constantly reassign priority to each task to determine
      which task will execute next. The event engine, upon receipt of the
      event **notification**, first passes **notification** to the
      registered destination with the highest **notification** level,
      followed by the next and so on. An option exists for the engine to
start
      with the event destination of the lowest **notification** level
      and continue until the highest level event destination has been
      serviced. If a broadcast mode of **notification** is desired, or
      if ordering is not important, then **registering** the event
      destination with any **notification** level will produce the
      desired results for broadcast events.

DETD FIG. 2 illustrates a method for managing event **notification** in
      a computer system (such as one or more of the systems 10, 16, 20 shown
      in FIG. 1) according to the present invention. The method includes a
      step 40 of **registering** at least one event consumer with a
      registry capable of holding multiple registrations. Registration
      is described in greater detail hereafter, but generally creates a
record
      of which event consumers to notify in response to which events.
DETD Suitable event consumers include event handlers of the kind familiar to
      those of skill in the art. One suitable **registry** includes a
      linked list of Event Control Blocks ("ECBs") such as the list of
      NESL.sub.-- ECB structures described in English and in the C
programming
      language in the '214 application; said description is hereby
      incorporated by reference. In addition to providing the **registry**
      with information such as an identification of the event handler and the
      event(s) of which it wishes **notification**, the event handler
      registration step 40 defines a registration order for the registered

event consumers

DETD   S